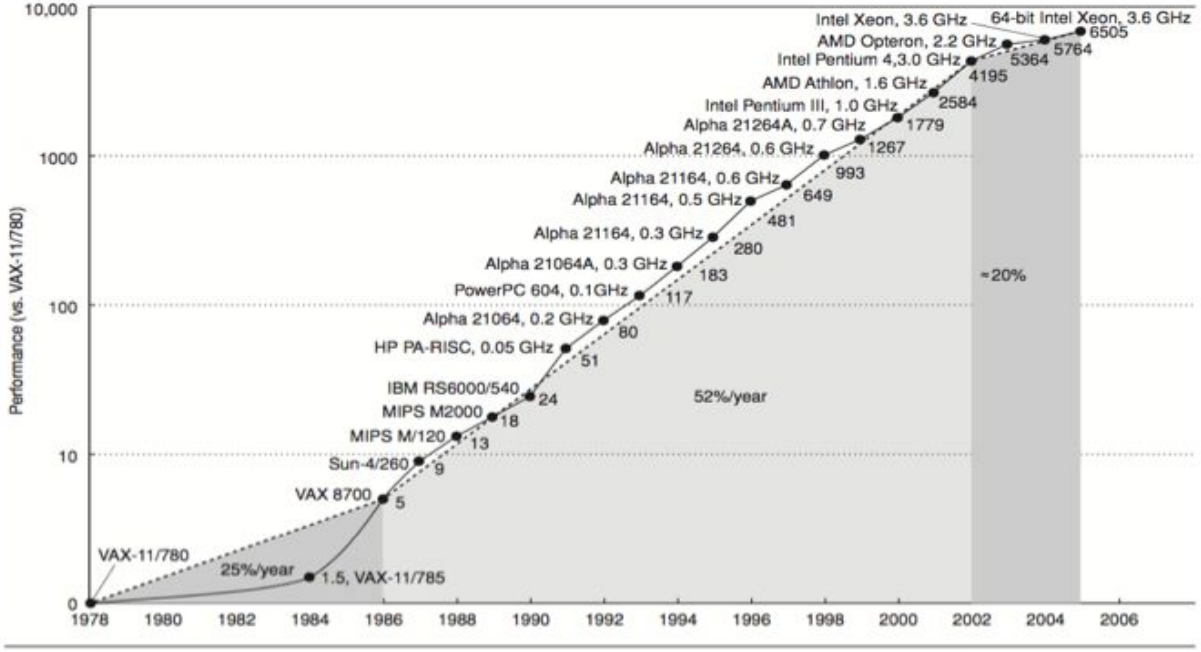
Implementing GCC5'S Profile-based optimization on Embedded Systems Using The Yocto Project

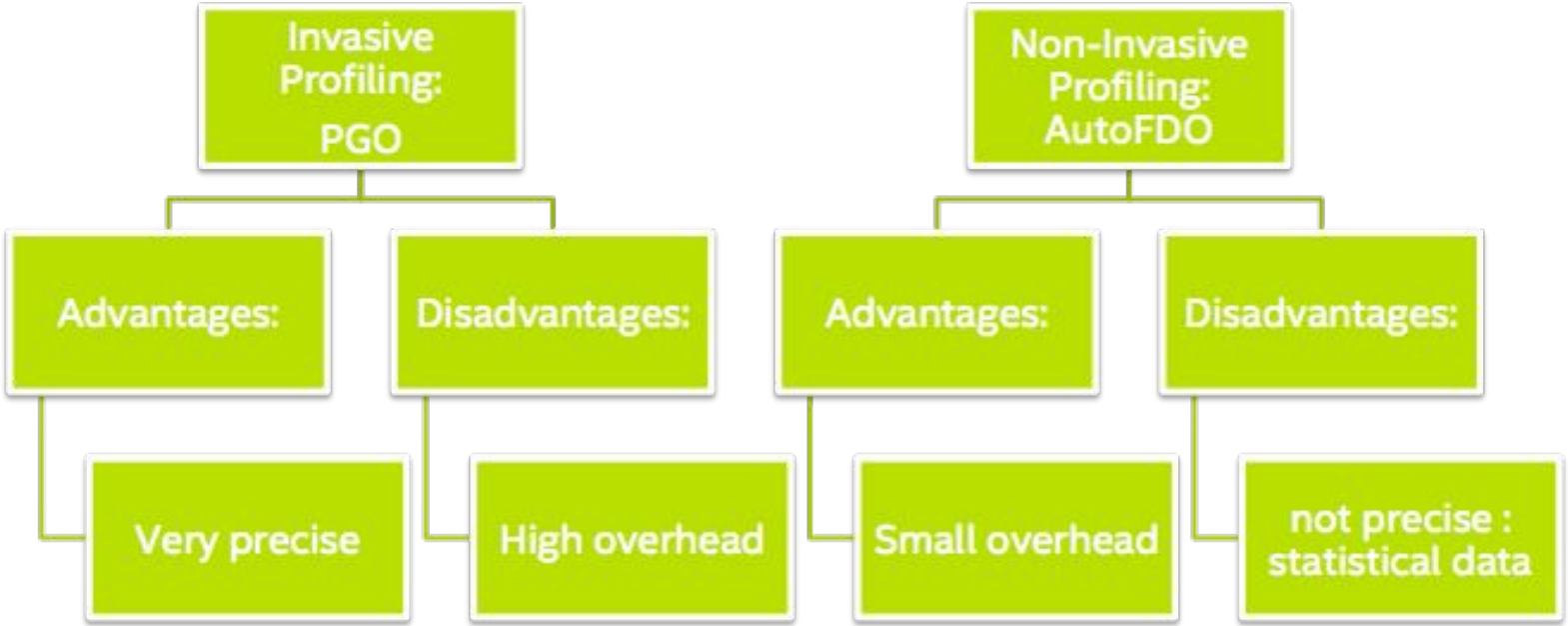
Embedded Linux Conference 2016, San Diego, CA
M. S. Alejandro Enedino Hernandez Samaniego
Linux fan: Victor Rodriguez

History of Computing Power...



Computer Architecture, Fifth Edition: A Quantitative Approach, Hennessy, John L. and Patterson, David A.

Profiling an application



AutoFDO Implementation

High level process

- Deploy application
- Run typical usage and use perf to collect profile (events)
- Create gcov file from perf.data
- Use gcov file to optimize application
- Deploy optimized application

Objective

Make it easy for Developers to include optimized applications on their devices.

- Seamless and non-invasive process
- Simple steps
- Automated
- Require little interaction from the developer

Initial Implementation

On Embedded Image

- Pmu-tools (ocperf)
- Autofdo
- Wrapper

Initial Implementation

On Development System:

recipes/matrixmult_1.0.bb

```
SUMMARY = "Matrix Multiplication test recipe to be implemented using AutoFDO"

inherit autotools autofdo
```

conf/local.conf

```
# Profile Application
AUTOFDO_FLAG = "0"
CORE_IMAGE_EXTRA_INSTALL += "matrixmult"
```

```
$ bitbake core-image-minimal
```

Initial Implementation

On Target:

```
# autofdo_matrixmult
Profiling application...

You can now use matrixmult.gcov to optimize matrixmult
```

- Transfer .gcov file to Dev System...

On Development System:

conf/local.conf

```
# Deploy Application
AUTOFDO_FLAG = "1"
CORE_IMAGE_EXTRA_INSTALL += "matrixmult"
```

```
$ bitbake core-image-minimal
```

Initial Drawbacks

- What if our application is written on a scripting language?
- Transferring the gcov file proves to be a burden
- Can we make it run automagically?

Implementation

- Inherit autofdo-profile class on our application’s recipe
- Inherit autofdo-compile class on what runs our application
- Create image using bitbake
- Run autofdo test using bitbake
- Rebuild application (and image) using bitbake

Implementation

On Development System:

recipes/matrixmult_1.0.bb

```
SUMMARY = "Matrix Multiplication test recipe to be implemented using AutoFDO"
inherit autotools autofdo-profile
```

recipes/python_2.7.11.bb

```
SUMMARY = "Python Language Recipe"

inherit autofdo-compile
```

conf/local.conf

```
# Profile Application
AUTOFDO_FLAG = "0"
AUTOFDO_BIN = “python”
AUTOFDO_APP = “matrixmult”
CORE_IMAGE_EXTRA_INSTALL += "matrixmult"
```

```
$ bitbake core-image-minimal
```

Implementation

On Development System:

conf/local.conf

```
INHERIT += "testimage"
TEST_TARGET = "simpleremote"
TEST_TARGET_IP = <TARGET IP ADDRESS>
TEST_SERVER_IP = <DEV SYSTEM IP ADDRESS>
TEST_SUITES =  " autofdo"
# Deploy Application
AUTOFDO_FLAG = “1”
```

Run test

```
$ bitbake core-image-minimal -c testimage
```

Deploy optimized application on target

```
$ bitbake core-image-minimal
```

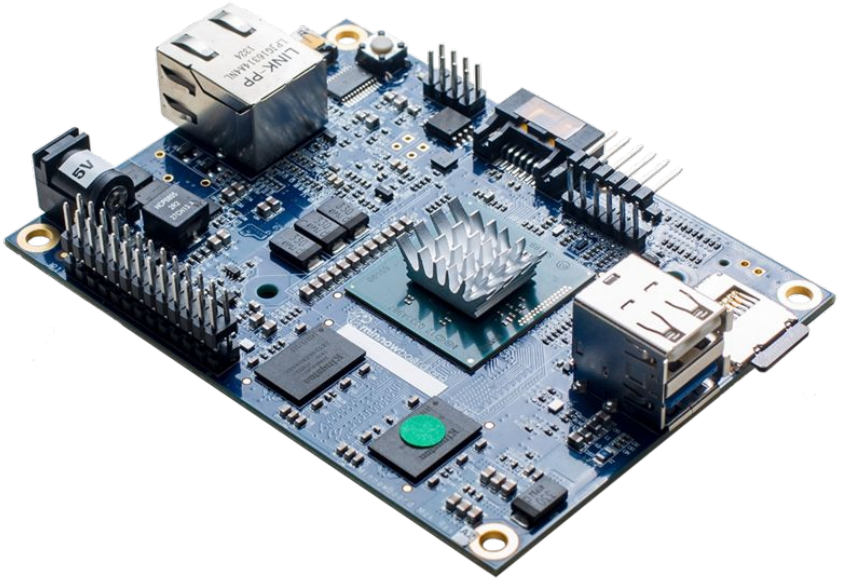
Results

Results

System Characteristics:

- 64-bit Intel®Atom™ E38xx Series SoC 1.33 GHz (dual-core)
- 2 GB DDR3 RAM System Memory

- Intel HD Graphics
- SATA 2
- USB 3.0
- Serial Debug FTDI
- Ethernet



Results

Execution Time Performance Table (ms)

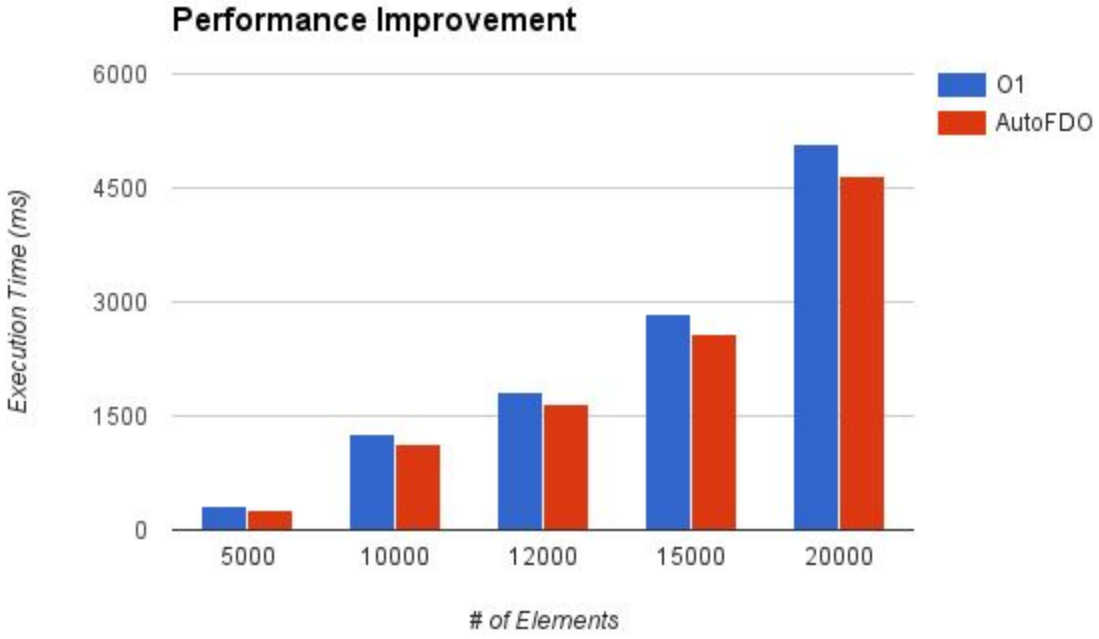
Elements	Original (ms)	O1 (ms)	AutoFDO (ms)
1000	57	12	10
5000	1433	306	272
10000	5701	1264	1144
12000	8154	1812	1649
15000	12787	2833	2588
20000	22936	5074	4666
30000	51211	11330	10473

Results

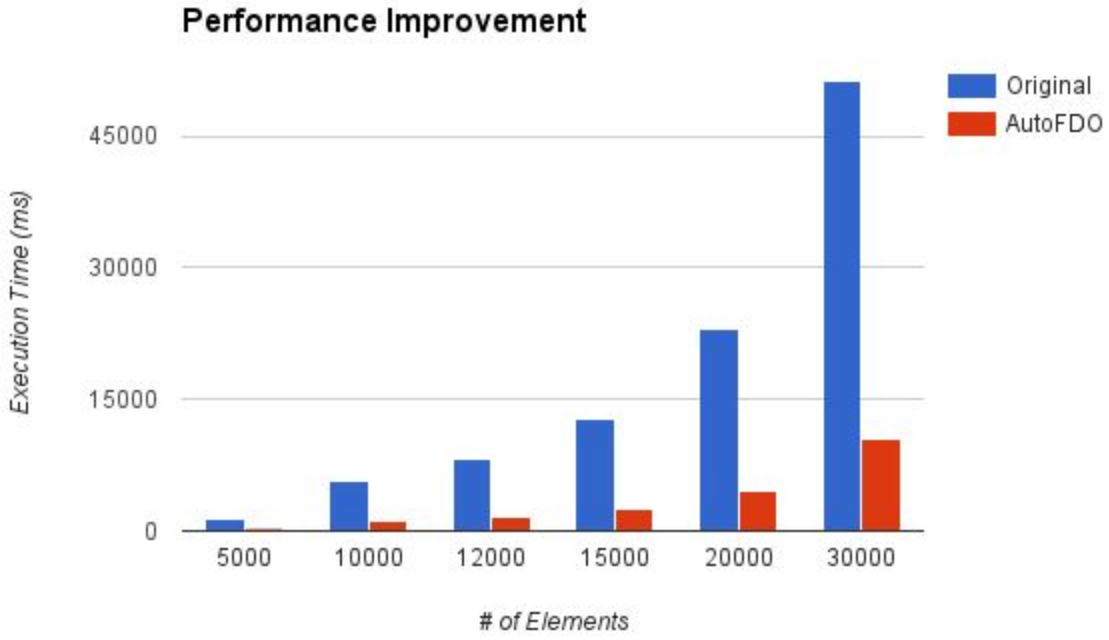
Performance Improvement Table (%)

Elements	Original vs O1	Original vs AutoFDO	O1 vs AutoFDO
1000	4.75	5.70	1.20
5000	4.68	5.27	1.13
10000	4.51	4.98	1.10
12000	4.50	4.94	1.10
15000	4.51	4.94	1.09
20000	4.52	4.92	1.09
30000	4.52	4.89	1.08

Results


Performance Improvement
6000
4500
3000
1500
0
Execution Time (ms)
5000
10000
12000
15000
20000
# of Elements
O1
AutoFDO

Results


Performance Improvement
Original
AutoFDO
45000
30000
15000
0
Execution Time (ms)
5000
10000
12000
15000
20000
30000
# of Elements

Future Work

Future Work

- Test on more applications and devices
  - E.g. Web servers
- Improve test case - Newest pmu-tools patches
- Make it “public”
- Able to provide optimized package using a package-manager

Help the compiler find the most efficient path...

Materials on Github:

http://github.com/aehs29/meta-autofdo

M. S. Alejandro Enedino Hernandez Samaniego
alejandro.hernandez@intel.com

Q & A

Thank You!

M. S. Alejandro Enedino Hernandez Samaniego
alejandro.hernandez@intel.com